

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-272756

(43)Date of publication of application : 18.10.1996

(51)Int.Cl. G06F 15/177
G06F 1/24
G06F 11/16

(21)Application number : 07-076291

(71)Applicant : TOSHIBA CORP

(22)Date of filing : 31.03.1995

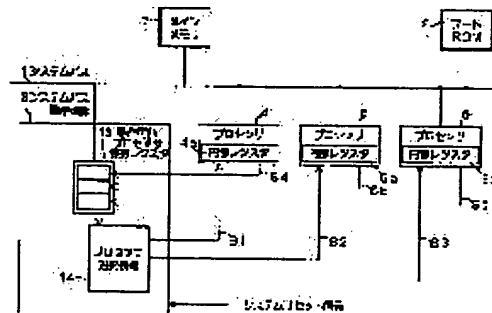
(72)Inventor : YAMAGAMI NOBUHIKO
NAKADA YASUMASA
UETOKO KATSUKI

(54) METHOD FOR STARTING MULTIPROCESSOR SYSTEM

(57)Abstract:

PURPOSE: To improve the reliability of the boot process of a multiprocessor system.

CONSTITUTION: The multiprocessor system which includes one boot ROM 2 storing codes for booting the system and allows plural processors 4, 5, and 6 to execute the codes stored in the boot ROM 2 is equipped with a system bus arbitrating mechanism 3 which selects one of processors outputting requests to acquire a system bus according to a rule prescribing the priority of the processors and gives system bus use permission at the request to acquire the bus outputted from all operable processors to fetch the codes in the boot ROM 2 when system resetting is reset, and does not give the system bus use permission to other processors until the one selected processor completes the execution of the codes stored in the boot ROM 2.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the
examiner's decision of rejection or application
converted registration]

[Date of final disposal for application]

BEST AVAILABLE COPY

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

* NOTICES *

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1. This document has been translated by computer. So the translation may not reflect the original precisely.
2. **** shows the word which can not be translated.
3. In the drawings, any words are not translated.

CLAIMS

[Claim(s)]

[Claim 1] In a multiprocessor system which has two or more processors which had the storage which memorized a bootstrap for starting a system, and were connected with said storage through a system bus When a system reset is canceled, based on a priority beforehand specified out of a processor which outputted an acquisition demand of said system bus, choose one processor, and licence of a system bus is given. A starting method of a multiprocessor system that said selected processor is characterized by not giving licence of a system bus at it until activation of said bootstrap is completed in a processor of others which were not chosen.

[Claim 2] In a multiprocessor system which has two or more processors which had the storage which memorized a bootstrap for starting a system, and were connected with said storage through a system bus As opposed to an acquisition demand of said system bus outputted from said two or more processors, respectively when a system reset is canceled A bootstrap which licence of a system bus was given one by one, and said two or more processors were alike, respectively, and was memorized more by said storage is performed. When one processor ends initialization routine of a processor of said bootstraps, while making the processor continue activation of said bootstrap A starting method of a multiprocessor system characterized by not performing future routines at them after ending initialization routine of a processor of said bootstraps in other processors.

[Claim 3] In a multiprocessor system which has two or more processors which had the storage which memorized a bootstrap for starting a system, and were connected with said storage through a system bus When a system reset is canceled, a bootstrap which chose one processor based on a priority beforehand specified out of said two or more processors, and was memorized by said storage is performed. When this selected processor ends initialization routine of a processor of said bootstraps in predetermined time When that processor is made to continue activation of said bootstrap and this selected processor does not end initialization routine of a processor of said bootstraps in predetermined time amount A starting method of a multiprocessor system characterized by performing a bootstrap which chose other processors based on said priority, and was memorized by said storage.

[Translation done.]

PAT-NO: JP408272756A

DOCUMENT-IDENTIFIER: JP 08272756 A

TITLE: METHOD FOR STARTING MULTIPROCESSOR SYSTEM

PUBN-DATE: October 18, 1996

INVENTOR-INFORMATION:

NAME

YAMAGAMI, NOBUHIKO

NAKADA, YASUMASA

UETOKO, KATSUKI

ASSIGNEE-INFORMATION:

NAME

TOSHIBA CORP

COUNTRY

N/A

APPL-NO: JP07076291

APPL-DATE: March 31, 1995

INT-CL (IPC): G06F015/177, G06F001/24, G06F011/16

ABSTRACT:

PURPOSE: To improve the reliability of the boot process of a multiprocessor system.

CONSTITUTION: The multiprocessor system which includes one boot ROM 2 storing codes for booting the system and allows plural processors 4, 5, and 6 to execute the codes stored in the boot ROM 2 is equipped with a system bus arbitrating mechanism 3 which selects one of processors outputting requests to acquire a system bus according to a rule prescribing the priority of the processors and gives system bus use permission at the request to acquire the bus outputted from all operable processors to fetch the codes in the boot ROM 2 when system resetting is reset, and does not give the system bus use permission to other processors until the one selected processor completes the execution of the codes stored in the boot ROM 2.

COPYRIGHT: (C)1996,JPO

"Yamagami"

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-272756

(43) 公開日 平成8年(1996)10月18日

(51) Int.Cl. ⁹	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/177			G 0 6 F 15/16	4 2 0 S
1/24			11/16	3 1 0 Z
11/16	3 1 0		1/00	3 5 0 B

審査請求 未請求 請求項の数 3 O L (全 12 頁)

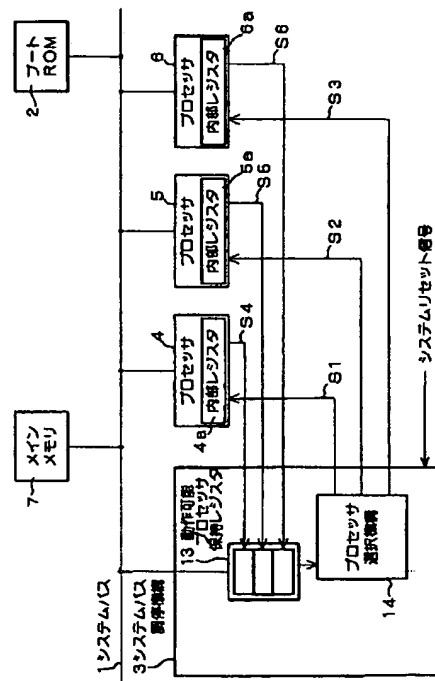
(21) 出願番号	特願平7-76291	(71) 出願人	000003078 株式会社東芝 神奈川県川崎市幸区堀川町72番地
(22) 出願日	平成7年(1995)3月31日	(72) 発明者	山上 宜彦 東京都青梅市末広町2丁目9番地 株式会 社東芝青梅工場内
		(72) 発明者	中田 恭正 東京都青梅市末広町2丁目9番地 株式会 社東芝青梅工場内
		(72) 発明者	上床 克樹 東京都青梅市末広町2丁目9番地 株式会 社東芝青梅工場内
		(74) 代理人	弁理士 鈴江 武彦

(54) 【発明の名称】 マルチプロセッサシステムの起動方法

(57) 【要約】

【目的】マルチプロセッサシステムのブートプロセスの際の信頼性を向上させることを可能にする。

【構成】システムをブートするためのコードが入ったブートROM 2が1つ存在し、複数のプロセッサ4, 5, 6がブートROM 2に収められていたコードを実行するマルチプロセッサシステムにおいて、システムリセットを解除したときに、動作可能な全てのプロセッサから、ブートROM 2中のコードをフェッチするために出力されるシステムバス獲得要求に対して、システムバス獲得要求を出力したプロセッサの中からプロセッサの優先順位を規定する規則に従って1つを選択してシステムバス使用許可を与え、その他のプロセッサには選択された1つのプロセッサによってブートROM 2に記憶されたコードの実行が完了するまでシステムバス使用許可を与えないシステムバス調停機構3を具備して構成する。



【特許請求の範囲】

【請求項1】 システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、

システムリセットが解除されたとき、前記システムバスの獲得要求を出力したプロセッサの中から予め規定された優先度に基づいて1つのプロセッサを選択してシステムバスの使用許可を与え、選択されなかったその他のプロセッサには、前記選択されたプロセッサが前記ブートストラップの実行が完了するまで、システムバスの使用許可を与えないことを特徴とするマルチプロセッサシステムの起動方法。

【請求項2】 システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、

システムリセットが解除されたとき、前記複数のプロセッサからそれぞれ出力される前記システムバスの獲得要求に対して、順次システムバスの使用許可を与え、前記複数のプロセッサのそれぞれにより前記記憶装置に記憶されたブートストラップを実行させ、1つのプロセッサが前記ブートストラップの内のプロセッサの初期化ルーチンを終了した際、そのプロセッサに前記ブートストラップの実行を継続させるとともに、その他のプロセッサには、前記ブートストラップの内のプロセッサの初期化ルーチンを終了した後、以後のルーチンを実行させないことを特徴とするマルチプロセッサシステムの起動方法。

【請求項3】 システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、

システムリセットが解除されたとき、前記複数のプロセッサの中から予め規定された優先度に基づいて1つのプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させ、この選択されたプロセッサが所定時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了した場合には、そのプロセッサに前記ブートストラップの実行を継続させ、この選択されたプロセッサが所定の時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了しなかった場合には、前記優先度に基づいてその他のプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させることを特徴とするマルチプロセッサシステムの起動方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、マルチプロセッサシステムにおける起動方法に関する。

【0002】

【従来の技術】 従来、マルチプロセッサシステムにおいては、複数のプロセッサによって共用されるシステムバスの使用を、システムバス調停機構（アービター）によって制御している。

【0003】 システムリセットを解除する場合には、全てのプロセッサに対するリセット信号が解除される。その結果、動作可能な全てのプロセッサは、システムを起動させるためのブートプロセスを実行するためのコード（プログラム）、すなわち、ブートストラップが格納されたブートROMの先頭番地の命令をフェッチするため、システムバス獲得要求をシステムバス調停機構に出力する。

【0004】 システムバス調停機構は、全てのプロセッサに対して、優先順位を規定するある規則に従って、システムバス獲得要求の単位でシステムバス使用許可を送出する。この場合、システムバス使用許可を受信したプロセッサは、システムバスを獲得して処理を実行するが、結果的に、全てのプロセッサがブートROMの中に格納されているプログラムのある一定の処理（例えばプロセッサ自身自身を初期化する処理）まで実行する。

【0005】 そして、ある一定の処理以降では、1つのプロセッサを除いて、無意味なダミー処理を実行する無限ループに入り、OS（オペレーティングシステム）によってマルチプロセッサ化されるのを待つ。また、1つのプロセッサは、その後のブートROMの中に格納されているプログラムを実行する。この処理には、ボード上の周辺回路などの初期化、I/Oデバイスの初期化、OSのローディングなどが含まれる。なお、ブートROM中に格納されたプログラムを継続して実行する1つのプロセッサは、予め決められている優先度に基づいて、最も優先度が高いスロットに装着されているボードのプロセッサに決定される。この優先度は具体的には、例えば、プロセッサが搭載されたボードを装置本体に装着するためのスロットに設定しておく。

【0006】

【発明が解決しようとする課題】 このように従来のマルチプロセッサシステムでは、ブートROMの中に格納されているプログラムを完全に継続して（ボード上の周辺回路などの初期化、I/Oデバイスの初期化、OSのローディングなどまで）実行する1つのプロセッサが予め決められていた。

【0007】 このため、その決められたプロセッサが故障していると、ブートプロセスが実行できず、システムが立ち上がらなくなってしまうという問題があった。すなわち、1つのプロセッサの故障だけで、他に正常なプロセッサが存在しているとしても、システム全体が使用できなくなってしまう場合があるため、ブートプロセスの信頼性を、より向上させることが望まれていた。

【0008】 本発明は前記のような事情を考慮してなさ

れたもので、マルチプロセッサシステムのブートプロセスの際の信頼性を向上させることが可能なマルチプロセッサシステムの起動方法を提供することを目的とする。

【0009】

【課題を解決するための手段】本発明は、システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、システムリセットが解除されたとき、前記システムバスの獲得要求を出力したプロセッサの中から予め規定された優先度に基づいて1つのプロセッサを選択してシステムバスの使用許可を与え、選択されなかったその他のプロセッサには、前記選択されたプロセッサが前記ブートストラップの実行が完了するまで、システムバスの使用許可を与えないことを特徴とする。

【0010】また、システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、システムリセットが解除されたとき、前記複数のプロセッサからそれぞれ出力される前記システムバスの獲得要求に対して、順次システムバスの使用許可を与え、前記複数のプロセッサのそれぞれにより前記記憶装置に記憶されたブートストラップを実行させ、1つのプロセッサが前記ブートストラップの内のプロセッサの初期化ルーチンを終了した際、そのプロセッサに前記ブートストラップの実行を継続させるとともに、その他のプロセッサには、前記ブートストラップの内のプロセッサの初期化ルーチンを終了した後、以後のルーチンを実行させないことを特徴とする。

【0011】また、システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、システムリセットが解除されたとき、前記複数のプロセッサの中から予め規定された優先度に基づいて1つのプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させ、この選択されたプロセッサが所定時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了した場合には、そのプロセッサに前記ブートストラップの実行を継続させ、この選択されたプロセッサが所定の時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了しなかった場合には、前記優先度に基づいてその他のプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させることを特徴とする。

【0012】

【作用】このような構成によれば、システムリセットが解除された際、ブートプロセスを実行するためにシステムバス獲得要求を出すことができる場合には、そのプロ

セッサは正常であるものとし、正常なプロセッサから1つを選択してブートプロセスを実行させる。従って、故障したプロセッサが存在し、そのプロセッサのブートプロセスを実行させる優先順位が最も高いとしても、正常なプロセッサが存在すれば、ブートプロセスを確実に実行させることができる。また、ブートプロセスの段階で正常と判別されたプロセッサを、マルチプロセッサ化の対象とすることができる。

【0013】また、システムリセットが解除された際、各プロセッサに対してブートプロセスを実行させ、ブートプロセスとして最初に実行されるプロセッサ自身を初期化するルーチンが終了できる場合には、そのプロセッサは正常であるものとし、正常なプロセッサから1つを選択して（最初に初期化するルーチンを完了したプロセッサ）、ブートプロセスをそれ以降も継続して実行させる。従って、システムバス獲得要求を出力できたものの、プロセッサ自身を初期化するルーチンが実行できない故障があるプロセッサが存在し、そのプロセッサのブートプロセスを実行させる優先順位が最も高いとしても、正常なプロセッサが存在すれば、ブートプロセスを確実に実行させることができる。

【0014】また、同様にしてプロセッサ自身を初期化するルーチンを完了できるか否かによって故障を判別するが、ブートプロセスを実行するプロセッサの優先順位を規定する規則に従って1つのプロセッサを選択し、正常なプロセッサが得られるまで、順次、対象とするプロセッサを変更してブートプロセスを試行する。従って、1つでも正常なプロセッサが存在すれば、ブートプロセスを実行させることができる。

【0015】

【実施例】以下、図面を参照して本発明の実施例を説明する。図1は本発明の第1実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図である。図1に示すように、第1実施例のマルチプロセッサシステムは、システムバス1を介して、ブートROM2、システムバス調停機構3、複数（本実施例では3つ）のプロセッサ4、5、6、及びメインメモリ7が相互に接続されて構成されている。

【0016】システムバス1は、プロセッサ4、5、6、ブートROM2、メインメモリ7、システムバス調停機構3などの間のデータのやり取りを行なう。ブートROM2は、ブートプロセスに必要なコード（プログラム）を格納するためのもので、プロセッサ自身の初期化、ボード上の周辺回路などの初期化、I/Oデバイスの初期化、OSのローディングなどの処理に関するプログラムが格納されている。

【0017】システムバス調停機構3は、プロセッサ4、5、6などからのシステムバス1に対する使用要求を調停するものである。システムバス調停機構3は、システムバス1と接続されている他、各プロセッサ4、

5、6との間でプロセッサ制御信号及びシステムバス調停機構制御信号の送受信を行なう。なお、プロセッサ制御信号は、システムバス調停機構3からプロセッサに対するリセット信号、システムバス使用許可信号を含んでいる。また、システムバス調停機構制御信号は、プロセッサからシステムバス調停機構3に対するシステムバス獲得要求信号を含んでいる。各信号が送受信される動作の詳細については後述する。

【0018】システムバス調停機構3には、動作可能プロセッサ保持レジスタ13、及びプロセッサ選択機構14が設けられている。動作可能プロセッサ保持レジスタ13は、システム内で動作可能なプロセッサを示す情報を各プロセッサ毎に保持するためのもので、システム内のプロセッサの数と同数のフィールドが設けられ、フィールドとプロセッサが対応づけられている。動作可能プロセッサ保持レジスタ13には、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。プロセッサ選択機構14は、動作可能プロセッサ保持レジスタ13に格納された情報を参照して、プロセッサからのシステムバス獲得要求信号に対して、動作可能なプロセッサのみにシステムバス使用許可信号を送出、あるいはシステムリセット信号に応じてプロセッサにプロセッサ制御信号のリセット信号を送出するものである。

【0019】プロセッサ4、5、6は、システムバス獲得要求信号をシステムバス調停機構3に送出的ることによって、システムバス1の使用権を獲得し、命令等を実行する。なお、本実施例の前提条件として、全てのプロセッサ4、5、6が持つ状態レジスタ、プログラムカウンタなど、プロセッサ4、5、6がそれぞれ動作するために必要な内部レジスタ4a、5a、6aには物理アドレスが割り当てられており、外部から（他のどのプロセッサからも）書き込みが可能であるとする。

【0020】メインメモリ7は、プロセッサ4、5、6によって実行されるコード（プログラム）やデータ等を格納するためのもので、ブート時には、ブートROM2に格納された最初に行われるプログラムによって、ブートROM2中のその他のブートプロセスのためのコードがブートROM2から読み出され格納される。また、メインメモリ7に格納されたブートプロセスのコードがプロセッサによって実行されて、OS（オペレーティングシステム）がローディングされる他、各種プログラムやデータ等が格納される。

【0021】次に、第1実施例の動作について、図2に示すフローチャートを参照しながら説明する。まず、システムバス調停機構3は、システムリセット信号がオンであるときには、プロセッサ選択機構14によって各プロセッサ4、5、6のそれぞれに対してプロセッサ制御信号S1、S2、S3のリセット信号を出力し、プロセッサにリセットをかけ続ける。その結果、プロセッサ

4、5、6は、命令フェッチなどの動作を全く行なわない。

【0022】システムリセット信号がオフとなりシステムリセットが解除されると、システムバス調停機構3は、プロセッサ制御信号S1、S2、S3のリセット信号を解除（オフ）することによって、プロセッサ4、5、6の動作を開始させる（ステップA1）。

【0023】この時、プロセッサ4、5、6は、ブートROM2の先頭番地の命令をフェッチするために、それぞれシステムバス調停機構制御信号S4、S5、S6のシステムバス獲得要求信号を用いて、システムバス調停機構3に対してシステムバス獲得要求を出す（ステップA2）。

【0024】システムバス調停機構3は、システムバス獲得要求を出したプロセッサが何れであるか、すなわちシステムバス獲得要求信号を正常に出力できた動作可能なプロセッサが何れであるかを動作可能プロセッサ保持レジスタ13に保持しておく（ステップA3）。

【0025】また、システムバス調停機構3は、プロセッサ選択機構14によって、動作可能プロセッサ保持レジスタ13に動作可能である情報が保持されたプロセッサの中から、優先順位を規定する規則に従って（プロセッサが搭載されたボードのスロットへの実装位置等）、最も優先度が高い1つのプロセッサを選択してシステムバス使用許可信号を出力する。また、システムバス調停機構3は、選択されなかった他のプロセッサに対してリセット信号を出力する（ステップA4）。

【0026】ここで、システムバス調停機構3からシステムバス使用許可信号を入力したプロセッサをプロセッサ4であるものとする。すなわち、プロセッサ選択機構14は、プロセッサ4にプロセッサ制御信号S1のシステムバス使用許可信号を用いてシステムバスの使用許可を与え、プロセッサ5、6にはプロセッサ制御信号S2、S3のリセット信号を用いて命令フェッチなどの動作を全く行なわないようにする。つまり、この状態では、動作するプロセッサはプロセッサ4だけとなる。

【0027】プロセッサ4は、システムバス1の使用権を獲得したので、システムバス1を介してコードを読み込んでブートプロセス、すなわちプロセッサ自身の初期化、ボード上の周辺回路などの初期化、I/Oデバイスの初期化などの処理を実行する（ステップA5）。

【0028】ブートプロセスが完了すると、プロセッサ4は、動作可能プロセッサ保持レジスタ13を読んで、動作可能なプロセッサが何れかを認識し、動作が可能なプロセッサに対して状態レジスタ、プログラムカウンタなどに適切な値を書き込んで、プロセッサ自身が初期化された状態にしてマルチプロセッサ化できる状態にする。また、システムバス調停機構3からのプロセッサ制御信号S2、S3のリセット信号を解除させて、プロセッサ5、6の動作を開始させる（ステップA6、A

7)。

【0029】前述した第1実施例において、障害を持つプロセッサを排除できることを説明する。プロセッサ4、5、6の中に障害を持つプロセッサがある場合には、まず、システムリセットが解除された際に、システムバス調停機構3にシステムバス獲得要求信号を出すことができない。

【0030】このため、障害があるためにシステムバス獲得要求信号が出力できなかったプロセッサについては、システムバス調停機構3の動作可能プロセッサ保持レジスタ13に正常であることを示す情報が設定されない。

【0031】従って、障害のあったプロセッサには、ブートプロセスを実行した1つのプロセッサにより、障害のあったプロセッサの状態レジスタ、プログラムカウンタなどの値が書き込まれず、そのプロセッサへのシステムバス調停機構3からのリセット信号が解除されないことになり、そのプロセッサを排除することができる。

【0032】このようにして、リセット信号が解除された際に、システムバス調停機構3に対してシステムバス獲得要求信号を出力できない故障のあったプロセッサが排除され、正常な他のプロセッサによってブートプロセスを実行させることができる。従って、ブートプロセスを実行すべき優先度の最も高い1つのプロセッサに故障があっても、他に正常なプロセッサが存在すれば、システムを起動させることができる。

【0033】次に、本発明の第2実施例について説明する。図3は本発明の第2実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図である。図3に示すように、システムバス1を介して、ブートROM2、システムバス調停機構23、複数（本実施例では3つ）のプロセッサ4、5、6、及びメインメモリ7が相互に接続されて構成されている。なお、図1に示す第1実施例のマルチプロセッサシステムと同一構成部分については同一符号を付して説明を省略する。

【0034】システムバス調停機構23は、プロセッサ4、5、6などからのシステムバス1に対する使用要求を調停するものである。システムバス調停機構23は、システムバス1と接続されている他、各プロセッサ4、5、6との間でプロセッサ制御信号及びシステムバス調停機構制御信号の送受信を行なう。なお、プロセッサ制御信号は、システムバス調停機構23からプロセッサに対するリセット信号、システムバス使用許可信号を含んでいる。また、システムバス調停機構制御信号は、プロセッサからシステムバス調停機構23に対するシステムバス獲得要求信号を含んでいる。各信号が送受信される動作の詳細については後述する。

【0035】システムバス調停機構23には、動作可能プロセッサ保持レジスタ33、プロセッサ選択機構34、及び初期化制御フラグ35が設けられている。動作

可能プロセッサ保持レジスタ33は、システム内で動作可能なプロセッサを示す情報を各プロセッサ毎に保持するためのもので、システム内のプロセッサの数と同数のフィールドが設けられ、フィールドとプロセッサが対応づけられている。動作可能プロセッサ保持レジスタ33には、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。プロセッサ選択機構34は、動作可能プロセッサ保持レジスタ33に格納された情報を参照して、プロセッサからのシステムバス獲得要求信号に対して、動作可能なプロセッサのみにシステムバス使用許可信号を送出、あるいはシステムリセット信号に応じてプロセッサにプロセッサ制御信号のリセット信号を送出するものである。初期化制御フラグ35は、複数のプロセッサの中で1つのプロセッサにブートプロセスを継続して実行させるために用いられる。初期化制御フラグ35は、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。

【0036】次に、第2実施例の動作について、図4に示すフローチャートを参照しながら説明する。まず、システムバス調停機構23は、システムリセット信号がオンであるときには、プロセッサ選択機構34によって各プロセッサ4、5、6のそれぞれに対してプロセッサ制御信号S1、S2、S3のリセット信号を出力し、プロセッサにリセットをかけ続ける。その結果、プロセッサ4、5、6は、命令フェッチなどの動作を全く行なわない。

【0037】システムリセット信号がオフとなりシステムリセットが解除されると、システムバス調停機構23は、プロセッサ制御信号S1、S2、S3のリセット信号を解除（オフ）することによって、プロセッサ4、5、6の動作を開始させる（ステップB1）。

【0038】この時、プロセッサ4、5、6は、ブートROM2の先頭番地の命令をフェッチするために、それぞれシステムバス調停機構制御信号S4、S5、S6のシステムバス獲得要求信号を用いて、システムバス調停機構23に対してシステムバス獲得要求を出す（ステップB2）。

【0039】システムバス調停機構23は、各プロセッサからのシステムバス獲得要求信号に対して、1つのプロセッサにシステムバス使用許可信号を出力し、システムバス1の使用権を与える（ステップB3）。

【0040】システムバス使用許可信号を入力したプロセッサは、ブートROM2のプログラム（コード）を読み込み、ブートプロセスの実行を開始する。ブートプロセスでは、まずプロセッサ自身を初期化する処理（初期化ルーチン）が実行される（ステップB4）。

【0041】システムバス調停機構23は、順次、システムバス獲得要求信号を出力したプロセッサにシステムバス使用許可を与える。このため、各プロセッサは、そ

れぞれプロセッサ自身を初期化する処理を実行する。

【0042】この結果、プロセッサ4、5、6の何れか1つが、最初にブートROM2に格納されたコード（プログラム）のプロセッサ自身を初期化するルーチンの実行を完了する（ステップB5）。ここでは、プロセッサ4が、最初に初期化ルーチンを完了したものとする。ブートROM2には、プロセッサ自身を初期化するルーチンが完了した後に、次のような処理を実行させるコードが格納されている。

【0043】すなわち、初期化ルーチンを最初に完了したプロセッサ4は、システムバス調停機構23の動作可能プロセッサ保持レジスタ33の自プロセッサに対応するフィールドに、初期化ルーチンが完了して正常であることを示すようにフラグを立てる（ステップB6）。 10

【0044】そして、プロセッサ4は、初期化制御フラグ35を参照し、フラグが立っていないならば、既に初期化ルーチンを完了したプロセッサが存在していることを示すようにフラグを立てる（ステップB7、B8）。プロセッサ4は、プロセッサ自身の初期化ルーチンに続くブートプロセス、すなわちボード上の周辺回路などの初期化、I/Oデバイスの初期化など、それ以降のコードの実行を続ける（ステップB9）。 20

【0045】一方、プロセッサ5、6は、初期化ルーチンを正常に完了できたら、プロセッサ4と同様にして、システムバス調停機構23の動作可能プロセッサ保持レジスタ33の自プロセッサに対応するフィールドにフラグを立てる（ステップB6）。

【0046】そして、プロセッサ5、6は、初期化制御フラグ35を参照するが、2番目移行にプロセッサ自身の初期化ルーチンを完了した場合、既にフラグが立っている（ステップB7）、初期化ルーチンを継続して実行せず、所定の状況となるまで実行される無意味な処理、すなわち無限ループを実行する（ステップB10）。例えば、無限ループには、OSによってマルチプロセッサ化される際に、分岐命令に書き換えられる対象となるダミー命令（nop命令）が含まれている。 30

【0047】プロセッサ4は、正常にブートプロセスを実行するので、OSをローディングし実行する。プロセッサ4は、OSのマルチプロセッサ化するための処理に基づいて、システムバス調停機構23の動作可能プロセッサ保持レジスタ33を参照して、動作可能な（プロセッサ自身の初期化が完了した）プロセッサを判別する。 40

【0048】プロセッサ4は、動作可能なプロセッサ、すなわちプロセッサ5、6が実行しているメインメモリ7中に格納されている無限ループ中のダミー命令を、マルチプロセッサ処理を実行する処理へ分岐する分岐命令に書き換えて、無限ループの処理から出して実質的に動作を開始させる。

【0049】なお、プロセッサ自身を初期化するルーチンを完了した2番目以降のプロセッサに対して、無限ル 50

ープの処理から出すためには、前述のような命令を書き換える方法の他に、無限ループによって所定回数以上の命令を実行した場合に、無限ループからマルチプロセッサ処理を実行する処理へ分岐するようになっていても良い。この場合、無限ループから出る前に、プロセッサ4によるブートプロセスが完了している必要がある。また、その他の方法であっても良い。

【0050】前述した第2実施例において、障害を持つプロセッサを排除できることを説明する。プロセッサ4、5、6の中に障害を持つプロセッサがある場合には、まず、そのプロセッサは、プロセッサ自身を初期化するルーチンを実行することができずに、動作可能プロセッサ保持レジスタ33に自プロセッサに対応するフィールドにフラグを立てる処理まで実行することができない。

【0051】従って、OSによってマルチプロセッサ化する際に、動作可能なプロセッサとは認識されないことになるので、そのプロセッサを排除することができる。このようにして、ブートROM2に格納されたプロセッサ自身を初期化するルーチンを実行できない故障のあったプロセッサが排除され、正常な他のプロセッサによってブートプロセスを実行させることができる。従って、ブートプロセスを実行すべき優先度の最も高い1つのプロセッサに故障があっても、他に正常なプロセッサが存在すれば、システムを起動させることができる。

【0052】次に、本発明の第3実施例について説明する。図5は本発明の第3実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図である。図5に示すように、システムバス1を介して、ブートROM2、システムバス調停機構43、複数（本実施例では3つ）のプロセッサ4、5、6、及びメインメモリ7が相互に接続されて構成されている。なお、図1に示す第1実施例のマルチプロセッサシステムと同一構成部分については同一符号を付して説明を省略する。

【0053】システムバス調停機構43は、プロセッサ4、5、6などからのシステムバス1に対する使用要求を調停するものである。システムバス調停機構43は、システムバス1と接続されている他、各プロセッサ4、5、6との間でプロセッサ制御信号及びシステムバス調停機構制御信号の送受信を行なう。なお、プロセッサ制御信号は、システムバス調停機構43からプロセッサに対するリセット信号、システムバス使用許可信号を含んでいる。また、システムバス調停機構制御信号は、プロセッサからシステムバス調停機構43に対するシステムバス獲得要求信号を含んでいる。各信号が送受信される動作の詳細については後述する。

【0054】システムバス調停機構43には、動作可能プロセッサ保持レジスタ53、プロセッサ制御部54、及びカウンタ55が設けられている。動作可能プロセッサ保持レジスタ53は、システム内で動作可能なプロセ 50

ッサを示す情報を各プロセッサ毎に保持するためのもので、システム内のプロセッサの数と同数のフィールドが設けられ、フィールドとプロセッサが対応づけられている。動作可能プロセッサ保持レジスタ53には、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。プロセッサ制御部54は、プロセッサからのシステムバス獲得要求信号に対して、動作可能なプロセッサにシステムバス使用許可信号を送出、あるいはシステムリセット信号に応じてプロセッサにプロセッサ制御信号のリセット信号を送出するものである。また、プロセッサ制御部54は、動作可能プロセッサ保持レジスタ53に格納された情報、及びカウンタ55からの通知に基づいて、動作可能なプロセッサによってブートプロセスを継続して実行させるものである。カウンタ55は、プロセッサがブートプロセスを開始してから所定の時間を計測するためのもので、プロセッサがブートROM2内のプロセッサ自身を初期化するルーチンの実行が終了するまでに要する時間に比べて十分に長い時間（例えば1.5倍～2倍程度）の計測を行なうものとする。例えば、カウンタ55は、システムクロックに同期するカウンタで、プロセッサ自身を初期化するルーチンの実行に要するサイクル数よりも十分に長いサイクル数をカウントできるものとする。カウンタ55は、所定の時間の計測を完了した場合にプロセッサ制御部54に通知する。

【0055】次に、第3実施例の動作について、図6に示すフローチャートを参照しながら説明する。まず、システムバス調停機構43は、システムリセット信号がオンであるときには、プロセッサ制御部54によって各プロセッサ4、5、6のそれぞれに対してプロセッサ制御信号S1、S2、S3のリセット信号を出力し、プロセッサにリセットをかけ続ける。その結果、プロセッサ4、5、6は、命令フェッチなどの動作を全く行なわない。

【0056】システムリセット信号がオフとなりシステムリセットが解除されると、システムバス調停機構43は、複数のプロセッサから優先順位を規定する規則に従って（プロセッサが搭載されたボードのスロットへの実装位置等）、最も優先度が高い1つのプロセッサを選択してリセット信号を解除（オフ）することによって動作を開始させる。システムバス調停機構43は、その他のプロセッサに対してはリセット信号を出力したままにしておく（ステップC1）。

【0057】ここで、システムバス調停機構43からリセット信号が解除されたプロセッサをプロセッサ4であるものとする。すなわち、プロセッサ制御部54は、プロセッサ4にプロセッサ制御信号S1のリセット信号を解除し、またシステムバス獲得要求信号に応じたシステムバス使用許可信号を用いてシステムバスの使用許可を与え、プロセッサ5、6にはプロセッサ制御信号S2、

S3のリセット信号を用いて命令フェッチなどの動作を全く行なわないようにしておく。

【0058】プロセッサ4は、システムバス1の使用権を獲得したので、システムバス1を介してコードを読み込んでブートプロセスの動作を開始する。ブートプロセスでは、まずプロセッサ自身を初期化する処理が実行される（ステップC2）。

【0059】一方、カウンタ55は、ブートプロセスの動作が開始されると同時に、時間の計測を開始する（ステップC3）。プロセッサ制御部54は、システムバス使用許可信号を出力したプロセッサ、すなわちブートプロセスのプロセッサ自身の初期化を実行しているプロセッサに対応する動作可能プロセッサ保持レジスタ53のフィールドに、フラグが設定されるかを監視する（ステップC4）。

【0060】ブートプロセスを実行するプロセッサ4は、ブートROM2内のコード（プログラム）のうち、プロセッサ自身を初期化するルーチンを終了すると、動作可能プロセッサ保持レジスタ53の自プロセッサ4に対応するフィールドに、正常であることを示すようにフラグを立てる（ステップC5）。

【0061】プロセッサ制御部54は、現在動作しているプロセッサ4のフィールドにフラグが設定されたことを検出すると、カウンタ55をリセットして、時間の計測を終了させる（ステップC6）。プロセッサ4は、そのままブートROM2内のプロセッサ自身の初期化ルーチンに続くブートプロセス、すなわちボード上の周辺回路などの初期化、I/Oデバイスの初期化などの実行を継続する（ステップC7）。

【0062】ブートプロセスを継続して実行させるプロセッサが得られると、プロセッサ制御部54は、その他のプロセッサ（プロセッサ5、6）に対して、順次、システムバス使用許可信号によってシステムバス1の使用を許可しながら、ブートプロセスを実行させる（ステップC8）。

【0063】2番目以降にブートプロセスを実行するプロセッサ5、6は、プロセッサ自身を初期化するルーチンを完了すると、システムバス調停機構43の動作可能プロセッサ保持レジスタ53の自プロセッサに対応するフィールドに、初期化ルーチンが完了して正常であることを示すようにフラグを立てる（ステップC9）。

【0064】プロセッサ5、6は、動作可能プロセッサ保持レジスタ53へのフラグの設定が完了すると、所定の状況となるまで実行される無意味な処理、すなわち無限ループを実行する。

【0065】無限ループの処理を実行するプロセッサ5、6は、前述した第2実施例と同様にして、正常にブートプロセスを実行したプロセッサ4によってメインメモリ7にローディングされたOSによって、マルチプロセッサ化される際に、命令が書き換えられるなどにより

実質的に動作を開始する。

【0066】一方、動作可能プロセッサ保持レジスタのフラグ53へのフラグの設定が、何らかの要因によって遅れた場合、もしくは不可能になった場合、カウンタ55は、所定の時間の計測を完了し、その旨をプロセッサ制御部54に通知する（ステップC10）。

【0067】プロセッサ制御部54は、カウンタ55からの通知を受けると、その時に動作中のプロセッサ（この場合はプロセッサ4）に対し、故障等の異常があるものとしてブートプロセスの実行を停止させるためにリセット信号を出力する（ステップC11）。

【0068】次に、プロセッサ制御部54は、プロセッサ4の次に優先度が高い1つのプロセッサ（プロセッサ5とする）を選択してシステムバス使用許可信号を出力し（リセット解除）、前述したプロセッサ4と同様にブートプロセス、すなわちプロセッサ自身を初期化する処理を実行させる（ステップC1、C2）。

【0069】この際、カウンタ55は、ブートプロセスを実行させるプロセッサが変更されたことによって、時間の計測を初期状態から開始する（ステップC3）。以下、前述と同様にして処理する。

【0070】すなわち、プロセッサにブートプロセスを実行させて、プロセッサ自身を初期化するルーチンが完了できなかった場合、順次、対象とするプロセッサを変更し、システム内の全てのプロセッサについて試行するまで繰り返して行なう。

【0071】このようにして、ブートプロセスを実行すべき1つのプロセッサを選択して、プロセッサ自身を初期化するルーチンを試行させ、正常に初期化処理が完了すればそのままブートプロセスを継続して実行させる。そして、プロセッサ自身を初期化するルーチンが所定の時間内に完了できなかった場合には、次のプロセッサを

選択して、同様にブートプロセス（初期化ルーチン）を試行する。従って、システム内に1つでも動作可能なプロセッサが存在すれば、ブートROM2内のコード（プログラム）を、プロセッサ自身を初期化するルーチンの後も継続して実行させることができ、システムを起動させることができる。

【0072】

【発明の効果】以上詳述したように本発明によれば、複数のプロセッサの中で故障したプロセッサが存在しても、正常な他のプロセッサによってブートプロセスを実行し、システムを起動することができるので、信頼性の高いマルチプロセッサシステムを構成することが可能となるものである。

【図面の簡単な説明】

【図1】本発明の第1実施例に係わるマルチプロセッサシステムの構成を示すブロック図。

【図2】第1実施例の動作を説明するためのフローチャート。

【図3】本発明の第2実施例に係わるマルチプロセッサシステムの構成を示すブロック図。

【図4】第2実施例の動作を説明するためのフローチャート。

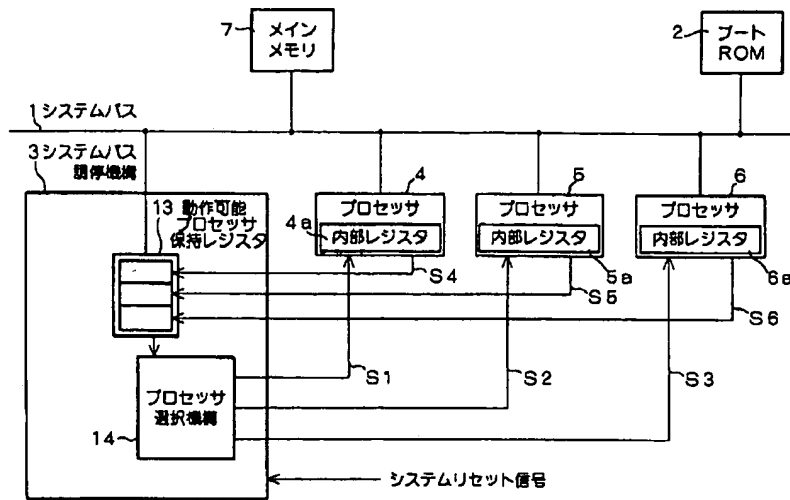
【図5】本発明の第3実施例に係わるマルチプロセッサシステムの構成を示すブロック図。

【図6】第3実施例の動作を説明するためのフローチャート。

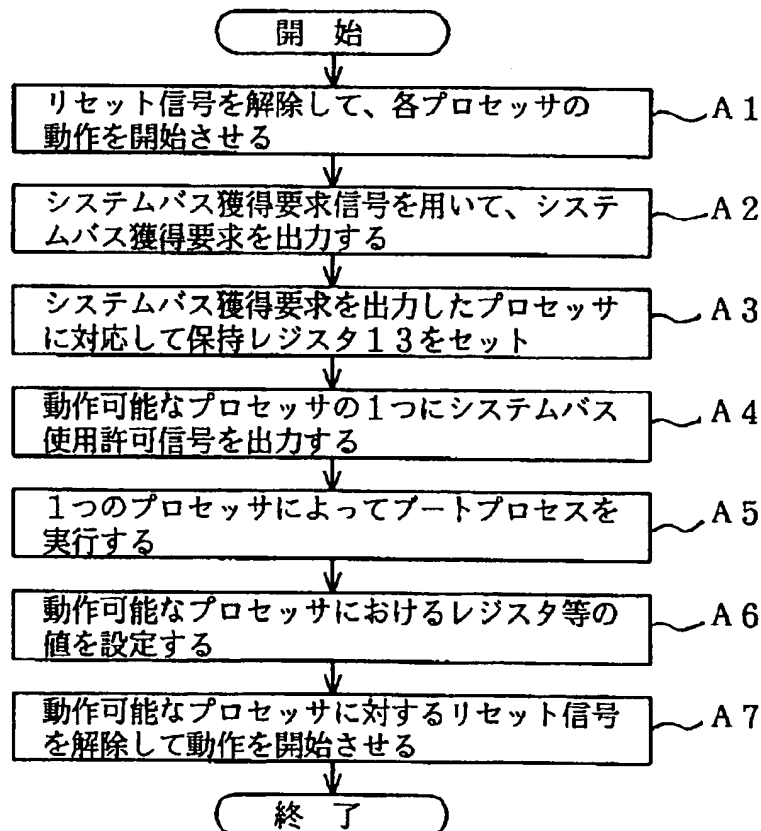
【符号の説明】

1…システムバス、2…ブートROM、3、23、43…システムバス調停機構、4、5、6…プロセッサ、7…メインメモリ、S1、S2、S3…プロセッサ制御信号、S4、S5、S6…システムバス調停機構制御信号。

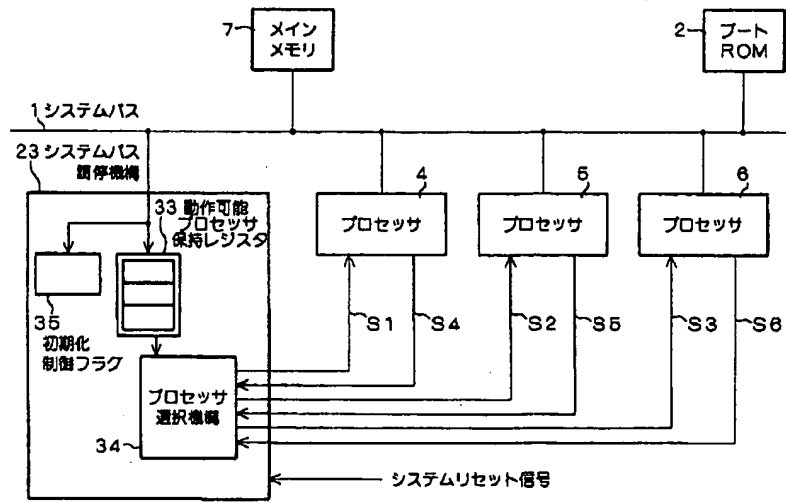
【図1】



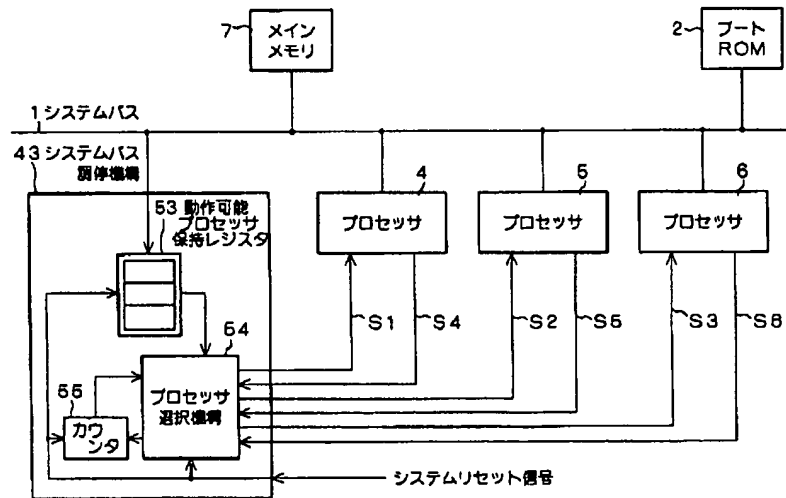
【図2】



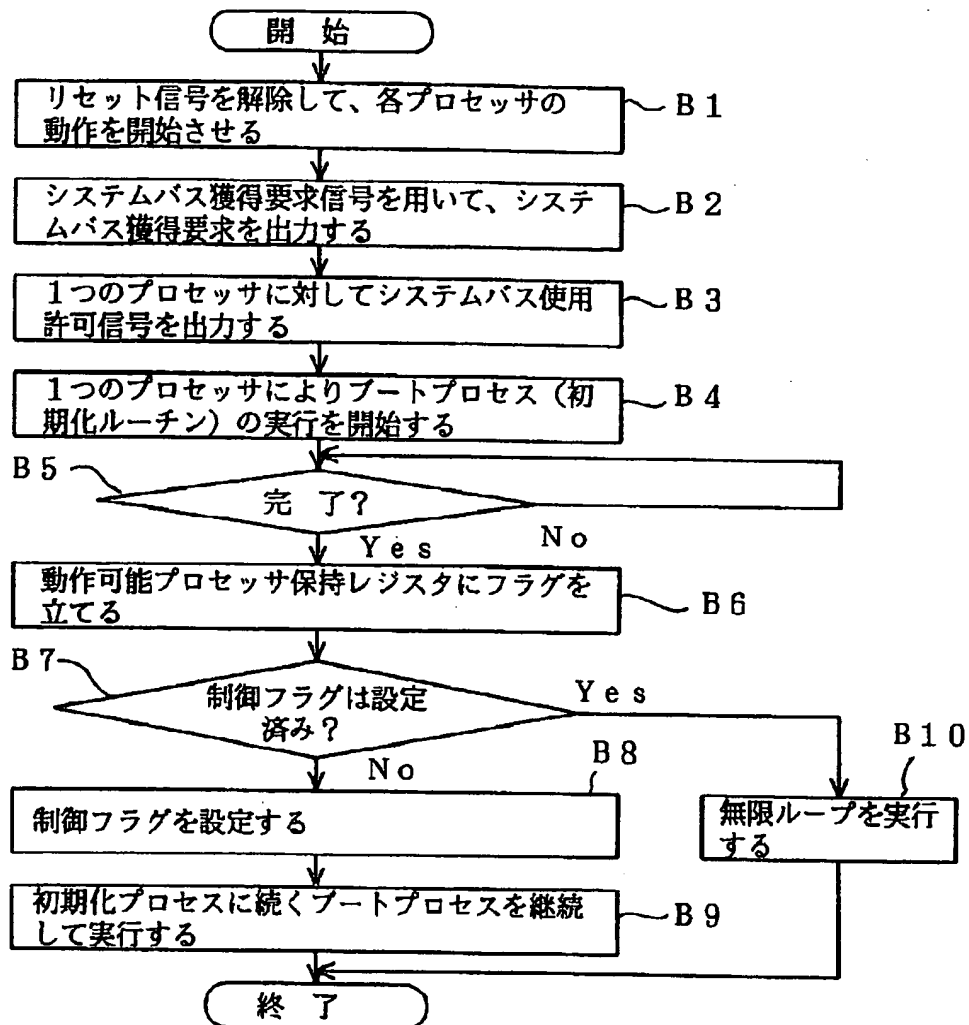
【図3】



【図5】



【図4】



【図6】

